

Intel® MPI Library for Linux* OS

Getting Started Guide

The Intel® MPI Library is a multi-fabric message passing library that implements the Message Passing Interface, version 2.1 (MPI-2.1) specification.

This *Getting Started Guide* explains how to use the Intel® MPI Library to compile and run a simple MPI program. This guide also includes basic usage examples and troubleshooting tips.

To quickly start using the Intel® MPI Library, print this short guide and walk through the example provided.

Copyright © 2003–2011 Intel Corporation

All Rights Reserved

Document Number: 315398-011

Revision: 4.0 Update 3

World Wide Web: <http://www.intel.com>

Contents

1	About this Document	5
1.1	Intended Audience	5
1.2	Using Doc Type Field.....	5
1.3	Conventions and Symbols.....	5
1.4	Related Information.....	6
2	Using the Intel® MPI Library.....	7
2.1	Usage Model	7
2.2	Before You Begin.....	7
2.3	Quick Start.....	7
2.4	Compiling and Linking	8
2.5	Setting up the Intel® MPI Library Environment.....	8
2.6	Selecting a Network Fabric	8
2.7	Running an MPI Program	10
3	Troubleshooting	12
3.1	Testing the Installation	12
3.2	Compiling and Running a Test Program	12
4	Next Steps	14

Disclaimer and Legal Notices

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm>

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to:

http://www.intel.com/products/processor_number/

MPEG-1, MPEG-2, MPEG-4, H.261, H.263, H.264, MP3, DV, VC-1, MJPEG, AC3, AAC, G.711, G.722, G.722.1, G.722.2, AMRWB, Extended AMRWB (AMRWB+), G.167, G.168, G.169, G.723.1, G.726, G.728, G.729, G.729.1, GSM AMR, GSM FR are international standards promoted by ISO, IEC, ITU, ETSI, 3GPP and other organizations. Implementations of these standards, or the standard enabled platforms may require licenses from various entities, including Intel Corporation.

BlueMoon, BunnyPeople, Celeron, Celeron Inside, Centrino, Centrino Inside, Cilk, Core Inside, E-GOLD, i960, Intel, the Intel logo, Intel AppUp, Intel Atom, Intel Atom Inside, Intel Core, Intel Inside, Intel Insider, the Intel Inside logo, Intel NetBurst, Intel NetMerge, Intel NetStructure, Intel SingleDriver, Intel SpeedStep, Intel Sponsors of Tomorrow., the Intel Sponsors of Tomorrow. logo, Intel StrataFlash, Intel vPro, Intel XScale, InTru, the InTru logo, the InTru Inside logo, InTru soundmark, Itanium, Itanium Inside, MCS, MMX, Moblin, Pentium, Pentium Inside, Puma, skool, the skool logo, SMARTi, Sound Mark, The Creators Project, The Journey Inside, Thunderbolt, Ultrabook, vPro Inside, VTune, Xeon, Xeon Inside, X-GOLD, XMM, X-PMU and XPOSYS are trademarks of Intel Corporation in the U.S. and other countries.

* Other names and brands may be claimed as the property of others.

Microsoft, Windows, Visual Studio, Visual C++, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Java is a registered trademark of Oracle and/or its affiliates.

Copyright (C) [2003]–[2011], Intel Corporation. All rights reserved.

Optimization Notice

Intel compilers, associated libraries and associated development tools may include or utilize options that optimize for instruction sets that are available in both Intel and non-Intel microprocessors (for example SIMD instruction sets), but do not optimize equally for non-Intel microprocessors. In addition, certain compiler options for Intel compilers, including some that are not specific to Intel micro-architecture, are reserved for Intel microprocessors. For a detailed description of Intel compiler options, including the instruction sets and specific microprocessors they implicate, please refer to the "Intel Compiler User and Reference Guides" under "Compiler Options." Many library routines that are part of Intel compiler products are more highly optimized for Intel microprocessors than for other microprocessors. While the compilers and libraries in Intel compiler products offer optimizations for both Intel and Intel-compatible microprocessors, depending on the options you select, your code and other factors, you likely will get extra performance on Intel microprocessors.

Intel compilers, associated libraries and associated development tools may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include Intel® Streaming SIMD Extensions 2 (Intel® SSE2), Intel® Streaming SIMD Extensions 3 (Intel® SSE3), and Supplemental Streaming SIMD Extensions 3 (SSSE3) instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors.

While Intel believes our compilers and libraries are excellent choices to assist in obtaining the best performance on Intel and non-Intel microprocessors, Intel recommends that you evaluate other compilers and libraries to determine which best meet your requirements. We hope to win your business by striving to offer the best performance of any compiler or library; please let us know if you find we do not.

Notice revision #20110307

1 About this Document

The *Intel® MPI Library for Linux* OS Getting Started Guide* contains information on the following subjects:

- First steps using the Intel® MPI Library
- First-aid troubleshooting actions

1.1 Intended Audience

This *Getting Started Guide* is intended for first time users.

1.2 Using Doc Type Field

This *Getting Started Guide* contains the following sections:

Document Organization

Section	Description
Section 1 About this Document	Section 1 introduces this document
Section 2 Using the Intel® MPI Library	Section 2 describes how to use the Intel® MPI Library
Section 3 Troubleshooting	Section 3 outlines first-aid troubleshooting actions
Section 4 Next Steps	Section 4 provides links to further resources

1.3 Conventions and Symbols

The following conventions are used in this document.

Table 1.3-1 Conventions and Symbols used in this Document

<i>This type style</i>	Document or product names
This type style	Hyperlinks
This type style	Commands, arguments, options, file names
THIS_TYPE_STYLE	Environment variables
<this type style>	Placeholders for actual values
[items]	Optional items
{ item item }	Selectable items separated by vertical bar(s)
(SDK only)	For Software Development Kit (SDK) users only

1.4 Related Information

To get more information about the Intel® MPI Library, see the following resources:

[Product Web Site](#)

[Intel® MPI Library Support](#)

[Intel® Cluster Tools Products](#)

[Intel® Software Development Products](#)

2 Using the Intel® MPI Library

2.1 Usage Model

Using the Intel® MPI Library involves the following steps:

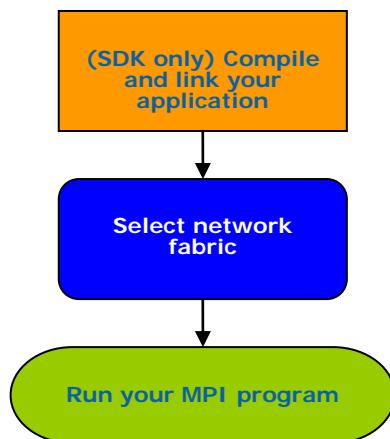


Figure 1: Flowchart representing the usage model for working with the Intel® MPI Library.

2.2 Before You Begin

Before using the Intel® MPI Library, ensure that the library, scripts, and utility applications are installed. See the product *Intel® MPI Library for Linux* OS Installation Guide* for installation instructions.

2.3 Quick Start

1. Source the `mpivars.[c]sh` script to establish the proper environment settings for the Intel® MPI Library. It is located in the `<installdir>/<arch>/bin` directory, where `<installdir>` refers to the Intel MPI Library installation directory (for example, `/opt/intel/impi`) and `<arch>` is one of the following architectures:
 - `ia32` – IA-32 architecture binaries
 - `intel64` – Intel® 64 architecture binaries.
2. Create an `mpd.hosts` text file that lists the nodes in the cluster using one host name per line.
3. **(SDK only)** Make sure you have a compiler in your `PATH`. To find the path to your compiler, run the `which` command on the desired compiler. For example:


```
$ which icc
/opt/intel/composerxe-2011/bin/intel64/icc
```
4. **(SDK only)** Compile a test program using the appropriate compiler driver. For example:


```
$ mpiicc -o myprog <installdir>/test/test.c
```

- Execute the test using the `mpirun` command.

```
$ mpirun -n <# of processes> ./myprog
```
- See the rest of this document and the *Intel® MPI Library Reference Manual* for more details.

2.4 Compiling and Linking

(SDK only)

To compile and link an MPI program with the Intel® MPI Library:

- Ensure that the underlying compiler and related software appear in your `PATH`.
 If you are using the Intel® Composer XE packages, ensure that the compiler library directories appear in the `LD_LIBRARY_PATH` environment variable.
 For example, for Intel® Composer XE 2011, execute the appropriate setup scripts:

```
/opt/intel/composerxe-2011/bin/compilervars.[c]sh
```
- Compile your MPI program by the appropriate `mpi` compiler script.
 For example, use the `mpicc` command to compile C code using the GNU* C compiler as follows:

```
$ mpicc -o myprog <installdir>/test/test.c
```

 where `<installdir>` is the full path to the installed package.

All supported compilers have equivalent commands that use the prefix `mpi` for the standard compiler command. For example, the Intel MPI Library command for the Intel® Fortran Compiler (`ifort`) is `mpiifort`.

2.5 Setting up the Intel® MPI Library Environment

The Intel® MPI Library uses the Hydra process manager. To run programs compiled with the `mpicc` (or related) commands, make sure your environment is set up correctly.

- Set up the environment variables with appropriate values and directories. For example, in the `.cshrc` or `.bashrc` files:
 - Ensure that the `PATH` variable includes the `<installdir>/<arch>/bin` directory. Use the `mpivars.[c]sh` scripts included with the Intel MPI Library to set up this variable.
 - (SDK only)** If you are using the Intel® Composer XE packages, ensure that the `LD_LIBRARY_PATH` variable contains the directories for the compiler library. Set this variable by using the `compilervars.[c]sh` scripts included with the compiler.
 - Set any additional environment variables that your application uses.
- Make sure that every node, rather than only one of them, can connect to any other node without a password.
- Create an `mpd.hosts` text file that lists the nodes in the cluster using one host name per line.
 For example:

```
$ cat > mpd.hosts
node1
node2
...
<ctrl>-D
```

2.6 Selecting a Network Fabric

The Intel® MPI Library dynamically selects the most appropriate fabric for communication between MPI processes. To select a specific fabric combination, set the new `I_MPI_FABRICS` or the old `I_MPI_DEVICE` environment variable.

I_MPI_FABRICS

(I_MPI_DEVICE)

Select a particular network fabric to be used for communication.

Syntax

```
I_MPI_FABRICS=<fabric>/<intra-node fabric>:<inter-node fabric>
```

Where *<fabric>* := {shm, dapl, tcp, tmi, ofa}

<intra-node fabric> := {shm, dapl, tcp, tmi, ofa}

<inter-nodes fabric> := {dapl, tcp, tmi, ofa}

Deprecated Syntax

```
I_MPI_DEVICE=<device>[:<provider>]
```

Arguments

<i><fabric></i>	Define a network fabric
shm	Shared-memory
dapl	DAPL-capable network fabrics such as InfiniBand*, iWarp*, Dolphin*, and XPMEM* (through DAPL*)
tcp	TCP/IP-capable network fabrics, such as Ethernet and InfiniBand* (through IPoIB*)
tmi	Network fabrics with tag matching capabilities through the Tag Matching Interface (TMI), such as Qlogic* and Myrinet*
ofa	Network fabric, such as InfiniBand* (through OpenFabrics* Enterprise Distribution (OFED*) verbs) provided by the Open Fabrics Alliance* (OFA*)

Correspondence with I_MPI_DEVICE

<i><device></i>	Equivalent notation for the I_MPI_FABRICS variable
sock	tcp
shm	shm
ssm	shm:tcp
rdma	dapl
rdssm	shm:dapl is the default value
<i><provider></i>	Optional DAPL* provider name (only for the rdma and rdssm devices) I_MPI_DAPL_PROVIDER= <i><provider></i> or I_MPI_DAPL_UD_PROVIDER= <i><provider></i>

Use the *<provider>* specification only for the {rdma, rdssm} devices.

For example, to select the OFED* InfiniBand* device, use the following command:

```
$ mpirun -n <# of processes> \
    -env I_MPI_FABRICS shm:dapl <executable>
```

For these devices, if `<provider>` is not specified, the first DAPL* provider in the `/etc/dat.conf` file is used.

Ensure that the selected fabric is available. For example, use `shm` only when all the processes can communicate with each other through the availability of the `/dev/shm` device. Use `dapl` only when all processes can communicate with each other through a single DAPL provider.

The `shm` fabric is available for both Intel® and non-Intel microprocessors, but it may perform additional optimizations for Intel microprocessors than it performs for non-Intel microprocessors.

2.7 Running an MPI Program

To launch programs linked with the Intel® MPI Library, use the `mpirun` command, as follows:

```
$ mpirun -n <# of processes> ./myprog
```

This command invokes the `mpiexec.hydra` command. Use the `mpiexec.hydra` options on the `mpirun` command line.

Use the `-n` option to set the number of MPI processes. This is the only obligatory option for the `mpirun` command.

If you are using a network fabric different than the default fabric, use the `-genv` option to assign a value to the `I_MPI_FABRICS` variable.

For example, to run an MPI program over the `shm` fabric, use the following command:

```
$ mpirun -genv I_MPI_FABRICS shm -n <# of processes> ./myprog
```

or

```
$ mpirun -genv I_MPI_DEVICE shm -n <# of processes> ./myprog
```

For a `dapl`-capable fabric, use the following command:

```
$ mpirun -genv I_MPI_FABRICS dapl -n <# of processes> ./myprog
```

or

```
$ mpirun -genv I_MPI_DEVICE rdma -n <# of processes> ./myprog
```

To use shared memory for intra-node communication and the DAPL layer for inter-node communication, use the following command:

```
$ mpirun -genv I_MPI_FABRICS shm:dapl -n <# of processes> ./myprog
```

or

```
$ mpirun -genv I_MPI_DEVICE rdssm -n <# of processes> ./myprog
```

or simply

```
$ mpirun -n <# of processes> ./myprog
```

To use shared memory for intra-node communication and TMI for inter-node communication, use the following command:

```
$ mpiexec.hydra -genv I_MPI_FABRICS shm:tmi -n <# of processes> ./myprog
```

Make sure that you have `libtmi.so` library in the search path of the `ldd` command. There is no way to select this fabric combination through the deprecated `I_MPI_DEVICE` variable.

To select shared memory for intra-node communication and OFED verbs for inter-node communication, use the following command:

```
$ mpirun -genv I_MPI_FABRICS shm:ofa -n <# of processes> ./myprog
```

There is no way to select this fabric combination through the deprecated `I_MPI_DEVICE` variable.

Set the `I_MPI_OFA_NUM_ADAPTERS` or the `I_MPI_OFA_NUM_PORT` environment variable to utilize the multitenant capabilities.

The exact settings depend on your cluster configuration. For example, if you have two InfiniBand* cards installed on your cluster nodes, use the following command:

```
$ export I_MPI_OFA_NUM_ADAPTERS=2
```

```
$ mpirun -genv I_MPI_FABRICS shm:ofa -n <# of processes> ./myprog
```

Set the `I_MPI_DAPL_UD` environment variable to enable connectionless DAPL User Datagrams (DAPL UD).

```
$ export I_MPI_DAPL_UD=enable
```

```
$ mpirun -genv I_MPI_FABRICS shm:dapl -n <# of processes> ./myprog
```

If you successfully run your application using the Intel MPI Library over any of the fabrics described, you can move your application from one cluster to another and use different fabrics between the nodes without re-linking. If you encounter problems, see [Troubleshooting](#) for possible solutions.

Additionally, using `mpirun` is the recommended practice when using a resource manager, such as PBS Pro* or LSF*.

For example, to run the application in the PBS environment, follow these steps:

1. Create a PBS launch script that specifies number of nodes requested, sets your Intel MPI Library environment, etc. For example, create a `pbs_run.sh` file with the following content:

```
#PBS -l nodes=2:ppn=1
#PBS -l walltime=1:30:00
#PBS -q workq
#PBS -V
# Set Intel MPI environment
mpi_dir=<installdir>/<arch>/bin
cd $PBS_O_WORKDIR
source $mpi_dir/mpivars.sh
# Launch application
mpirun -n <# of processes> ./myprog
```

2. Submit the job using the PBS `qsub` command:

```
$ qsub pbs_run.sh
```

When using `mpirun` under a job scheduler, you do not need to determine the number of available nodes. Intel MPI Library automatically detects the available nodes through the Hydra process manager.

3 Troubleshooting

Use the following sections to troubleshoot problems with installation, setup, and execution of applications using the Intel® MPI Library.

3.1 Testing the Installation

To ensure that the Intel® MPI Library is installed and functioning correctly, complete the general testing below, in addition to compiling and running a test program.

To test the installation (on each node of your cluster):

1. Verify that `<installdir>/<arch>/bin` is in your `PATH`:

```
$ ssh <nodename> which mpirun
```

You should see the correct path for each node you test.

(SDK only) If you use the Intel® Composer XE packages, verify that the appropriate directories are included in the `PATH` and `LD_LIBRARY_PATH` environment variables

```
$ mpirun -n <# of processes> env | grep PATH
```

You should see the correct directories for these path variables for each node you test. If not, call the appropriate `compilervars.[c]sh` script. For example, for the Intel® Composer XE 2011 use the following source command:

```
$ ./opt/intel/composerxe-2011/bin/compilervars.sh
```

2. In some unusual circumstances, you need to include the `<installdir>/<arch>/lib` directory in your `LD_LIBRARY_PATH`. To verify your `LD_LIBRARY_PATH` settings, use the command:

```
$ mpirun -n <# of processes> env | grep LD_LIBRARY_PATH
```

3.2 Compiling and Running a Test Program

To compile and run a test program, do the following:

1. **(SDK only)** Compile one of the test programs included with the product release as follows:

```
$ cd <installdir>/test
```

```
$ mpiicc -o myprog test.c
```

2. If you are using InfiniBand*, Myrinet*, or other RDMA-capable network hardware and software, verify that everything is functioning correctly using the testing facilities of the respective network.
3. Run the test program with all available configurations on your cluster.

- Test the TCP/IP-capable network fabric using:

```
$ mpirun -n 2 -env I_MPI_DEBUG 2 -env I_MPI_FABRICS tcp ./myprog
```

You should see one line of output for each rank, as well as debug output indicating the TCP/IP-capable network fabric is being used.

- Test the shared-memory and DAPL-capable network fabrics using:

```
$ mpirun -n 2 -env I_MPI_DEBUG 2 -env I_MPI_FABRICS shm:dapl ./myprog
```

You should see one line of output for each rank, as well as debug output indicating the shared-memory and DAPL-capable network fabrics are being used.

- Test any other fabric using:

```
$ mpirun -n 2 -env I_MPI_DEBUG 2 -env I_MPI_FABRICS <fabric> ./myprog
```

where `<fabric>` can be a supported fabric. For more information, see [Selecting a Network Fabric](#).

For each of the `mpirun` commands used, you should see one line of output for each rank, as well as debug output indicating which fabric was being used. The fabric(s) should agree with the `I_MPI_FABRICS` setting.

The `<installdir>/test` directory in the Intel® MPI Library Development Kit contains other test programs in addition to `test.c`.

4 *Next Steps*

To get more information about the Intel® MPI Library, explore the following resources:

See the *Intel® MPI Library Release Notes* for updated information on requirements, technical support, and known limitations.

The *Intel® MPI Library Reference Manual* for in-depth knowledge of the product features, commands, options, and environment variables.

Visit the [Intel® MPI Library for Linux* Knowledge Base](#) for additional troubleshooting tips and tricks, compatibility notes, known issues, and technical notes.

For more information see Websites:

[Product Web Site](#)

[Intel® MPI Library Support](#)

[Intel® Cluster Tools Products](#)

[Intel® Software Development Products](#)